一种基于单纯形搜索的粒子群优化算法 *

胡锦帆，张晓伟†，袁岐江，张为军，程崇东

(电子科技大学 数学科学学院，成都 611731)

摘　要：为了改善粒子群优化算法的求解性能，提出了一种基于单纯形搜索和粒子群优化的混合算法。该算法一方面自适应地确定惯性权重、认知以及社会参数来达到免参数目的，另一方面利用单纯形搜索来引导部分粒子的搜索方向，从而加速算法收敛。数值实验结果表明，与传统的粒子群算法和其他基于单纯形的粒子群算法相对比，该算法在评估次数、求解精度方面表现良好。

关键词：直接搜索；单纯形搜索；粒子群优化

中图分类号：TP301.6　　doi: 10.3969/j.issn.1001-3695.2018.06.0465

Particle swarm optimization based on simplex search

Hu Jinfan, Zhang Xiaowei†, Yuan Qijiang, Zhang Weijun, Cheng Chongdong

(*School of Mathematical Sciences*, *University of Electronic Science & Technology of China*, *Chengdu* 611731, *China*)

Abstract: In order to improve the performance of particle swarm optimization, the paper proposed a hybrid algorithm based on simplex search and particle swarm optimization. On the one hand, the algorithm determines the inertia weight adaptively, cognition and social parameters for the purpose of avoiding parameters. On the other hand, the simplex search is used to guide the direction of several particles, thus the convergence of the algorithm is accelerated. The results of numerical experiments show that the proposed algorithm has better performance than the compared algorithms in terms of function evaluations and accuracy.

Key words: direct search; simplex search; particle swarm optimization

0　引言

粒子群算法是由 Kennedy 和 Eberhart 于 1995 年提出，它通过模拟鸟群觅食过程中的迁徙和群聚行为来寻找全局最优解[1~3]。1998 年 Shi 等人引入了惯性权重机制，使得粒子群算法的性能得到进一步改善[4~8]。与传统优化方法相比，粒子群算法具有如下优点：不需要目标函数导数等解析信息，具有很好的通用性；不易陷入局部最优，具有较强的鲁棒性；流程简单，易于实现。然而，粒子群算法的启发式机制往往使得算法易于早熟，而且对于特定问题，算法性能过多依赖于对参数的设定。如今许多改进算法也未能很好解决参数设置过多，且参数设置依靠经验[9,10]，而对于简洁的无参数改进算法而言，算法易早熟收敛[11]。单纯形搜索能够利用问题本身较弱的解析性质，理论基础丰富，搜索具有针对性，收敛速度较快，并且和粒子群优化有着天然的共性，便于取长补短。考虑到这两个算法的特点，出现了许多基于单纯形的粒子群算法[12-14]。

近年来，有许多基于单纯形搜索的粒子群优化算法研究成果，并在工程领域到广泛应用[15~19]。2007 年，Fan 和 Zahira 提出了一种求解无约束优化问题的混合算法，该算法将单纯形搜索与粒子群算法相结合，以提高粒子群优化的收敛速率，其采用的算法结构为后续算法设计提供了思路，但该算法参数设置较多，且对较高维数的优化测试函数的求解性能有待进一步提高[16]。2008 年，Hsu 和 Gao 在粒子群优化中嵌入了单纯形搜索机制，该机制每隔 k 次粒子群优化算法迭代便执行 1 次单纯形搜索，从而不但可以减少函数评估次数，而且较好地平衡了搜索和探索之间的矛盾，但是数值实验部分几乎只评估了 2-5 维优化函数的求解性能[17]。2010 年，任小康等人提出了一种基于单纯形法的量子粒子群优化算法，该算法先用量子粒子群算法进行全局搜索，然后依据种群适应度标准差确定是否利用单纯形法进行局部搜索[18]，但是如何根据种群适应度标准差判断种群是否陷入“早熟”是个困难的事情，从而限制了算法的进一步推广。2018 年，武可栋等人在粒子群优化算法的初始化过程中加入了单纯形搜索，利用单纯形法对初始值进行预处理[19]，并针对 20 个 10 维测试函数进行了数值模拟，实验结果表明其

收稿日期：2018-06-26；修回日期：2018-08-31　基金项目：中央高校科研基本业务费资助项目(ZYGX2016J130)；国家自然科学基金资助项目(61471102)

作者简介：胡锦帆（1997-），男，湖南郴州人，主要研究方向为智能优化；张晓伟（1979-），男（通信作者），陕西宝鸡人，博士，主要研究方向为最优化方法、进化计算（zhangxiaowei@uestc.edu.cn）；袁岐江（1996-），男，辽宁丹东人，主要研究方向为智能优化；张为军（1997-），男，安徽宣城人，主要研究方向为智能优化；程崇东（2000-），男，江西南昌人，主要研究方向为智能优化.

相比标准粒子群优化算法具有更高的求解精度，但是该算法存在参数设置困难，评估次数较多的缺点。

为了改善粒子群优化算法的求解性能，本文提出了一种基于单纯形搜索的粒子群优化算法，该方法一方面利用类电磁机制原理[12]自适应地确定惯性权重、认知以及社会参数来达到免参目的，另一方面利用单纯形搜索来干预每次搜索方向及步长，从而加速算法收敛。

1　粒子群算法

粒子群算法（particle swarm optimization，PSO）的思想来源于对鸟群飞行或觅食过程中的迁徙和群集行为[1~3,13,14]。鸟仅仅是追踪它有限数量的邻居，但最终的整体结果是整个鸟群好像在一个中心的控制之下，即复杂的全局行为是由简单规则的相互作用引起的。不同于达尔文“适者生存，优胜劣汰”进化思想，粒子群优化算法是通过个体之间的协作来寻找最优解。

PSO 算法描述如下：

a)初始化种群 $P(t)$，速度 $V(t)$，认知系数 c_1，社会系数 c_2，惯性权重 w，令 $t=0$；

b)计算每一个粒子 $X_i \in P(t)$ 的适应值，个体经历最优位置 $g_i(t)$，种群全局最优位置 $G(t)$；

c) 更新每一个粒子 $X_i \in P(t)$ 的速度 $V_{i,j}(t+1)$ 和位置 $X_i(t+1)$：

for　$X_i(t) \in P(t)$

for　$j=1$　to n

$$V_{i,j}(t+1) = w \cdot V_{i,j}(t) + c_1 \cdot rand \cdot \left(g_{i,j}(t) - X_{i,j}(t)\right) + c_2 \cdot rand \cdot \left(G_j(t) - X_{i,j}(t)\right) \quad (1)$$

end

$$X_i(t+1) = X_i(t) + V_i(t+1) \quad (2)$$

end

d)若满足终止条件，停机；否则，$t=t+1$，转 b)。

这里，$V_{i,j}(t)$ 表示 $V_i(t)$ 的第 j 个分量，$X_{i,j}(t)$ 表示 $X_i(t)$ 的第 j 个分量，n 表示维数，即分量个数；c_1 和 c_2 是两个正常数；w 是惯性权重，$rand$ 是一个(0,1)之间的随机数。

在步骤 c)中，式（1）说明粒子的速度取决于上次的速度更新，它自己所到达的历史最佳位置和全种群中所有粒子的历史最佳位置;式（2）表明每个粒子的位置在搜索域中的更新方式。

2　单纯形搜索

单纯形（simplex）是代数拓扑中的基本概念，是三角形和四面体的一种泛化。n 维单纯形是一个由 n+1 个点组成的凸包。例如：1 维单纯形就是线段；2 维单纯形就是三角形；3 维单纯形就是一个四面体。

单纯形搜索（Nelder–mead simplex search，简称 NM）[20]通过反射、扩展、压缩等一系列初等几何变换操作来寻求问题的最优解。每次变换后，试图用一个更好的顶点替换当前已知最差点。

以最优化问题 $\min_{X \in R^2} f(X)$ 为例，分别计算 3 个顶点 X_{high}，X_{sec}，X_{low} 的函数值 f_{high}，f_{sec}，f_{low}。不妨假设，$f_{high} > f_{sec} > f_{low}$。基本操作(图 1~4)如下：

a）反射。取 X_{low} 和 X_{sec} 的中点 X_{cent}，计算反射点 $X_{ref} = X_{cent} + \left(X_{cent} - X_{high}\right)$。

b）扩展。若 $f_{low} > f_{ref}$，则计算扩展点 $X_{exp} = X_{cent} + 2*\left(X_{ref} - X_{cent}\right)$。

c）压缩。(a) 若 $f_{high} > f_{ref} \ge f_{sec}$，计算向外压缩点 $X_{cont} = X_{cent} + 0.5*\left(X_{ref} - X_{cent}\right)$

若 $f_{ref} > f_{cont}$，用 X_{cont} 取代 X_{high}；

(b) 若 $f_{ref} \ge f_{high}$，计算向内压缩点 $X_{cont} = X_{cent} + 0.5*\left(X_{high} - X_{cent}\right)$

若 $f_{high} > f_{cont}$，用 X_{cont} 取代 X_{high}。

d）收缩。若 $f_{cont} > f_{high}$，则向最好点 X_{low} 收缩其余顶点，即对 $\forall i$，$i \ne low$，$X_i = 0.5*(X_i + X_{low})$。

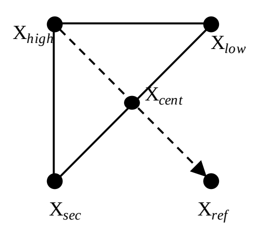


图 1　反射操作

Fig.1　Reflection

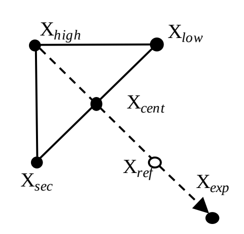
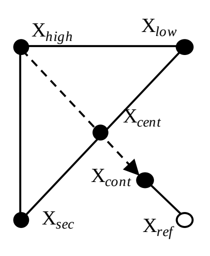


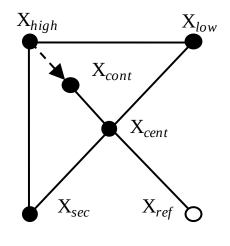
图 2　扩展操作

Fig.2　Expansion



(a) 向外压缩操作

(a) Outside contraction



(b) 向里压缩操作

(b) Inside contraction

图 3　压缩操作

Fig.3　Contraction

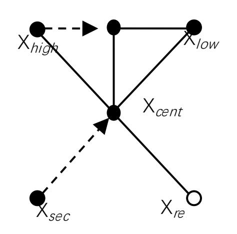


图 4 收缩操作

Fig.4 Shrink

NM 算法描述如下：

a)计算 n 维单纯形的 n+1 顶点 X_i 的函数值 f_i，$i=1,\cdots,n$，选取最差解 X_{high}，次差解 X_{sec}，最好解 X_{low}，中点 $X_{cent}=0.5*(X_{low}+X_{sec})$，及其对应函数值为 f_{high}，f_{sec}，f_{low}，f_{cent}。

b)计算反射点 $X_{ref}=X_{cent}+\left(X_{cent}-X_{high}\right)$ 以及对应函数值为 f_{ref}。

c)进行单纯形变换：

if f_{ref} > f_{low}

if $f_{ref}\geq f_{sec}$

计算压缩点 $X_{cont}=X_{cent}+0.5*(X_{high}-X_{cent})$

如果 f_{cont} < f_{high}，则 $X_{high}=X_{cont}$；否则向 X_{low} 进行收缩操作。

elseif $f_{high}>f_{ref}\geq f_{sec}$

$$X_{cont}=X_{cent}+0.5*\left(X_{ref}-X_{cent}\right)$$

如果 f_{cont} < f_{high}，则 $X_{high}=X_{cont}$；否则向 X_{low} 进行收缩操作。

end

$$X_{high}=X_{ref}$$

else

计算扩展点 $X_{\exp}=X_{cent}+2*\left(X_{ref}-X_{cent}\right)$，如果 f_{exp} < f_{refl}，则 $X_{high}=X_{exp}$；否则，$X_{high}=X_{ref}$。

end

通过步骤 c)的操作，新单纯形的 $n+1$ 个顶点中至少存在一个顶点进行了改变。

3 基于单纯形搜索的粒子群优化算法

3.1 PSO 算法改进

在传统粒子群优化中，第 i 个粒子的位置更新基于三个因素，见式（1）（2）。然而，在基于单纯形搜索的粒子群优化算法(Nelder-mead simplex particle swarm optimization，NM-PSO)中，对于第 t 代种群 $P(t)$ 中第 i 个粒子 $X_i(t)$，通过式(3)～(5) 更新：

$$V_i=F_0+F_1+F_2 \tag{3}$$

$$T_i(t)=X_{r3}(t)+V_i \tag{4}$$

$$X_i(t+1)=T_i(t) \quad \text{if} \quad f(T_i(t)) < f_i(t) \tag{5}$$

这里：

$$F_0=(X_{r1}(t)-X_{r3}(t))*[f(X_{r3}(t))-f(X_{r1}(t))]/D$$

$$F_1=(g_{r2}(t)-X_{r3}(t))*[f(X_{r3}(t))-f(g_{r2}(t))]/D$$

$$F_2=(G(t)-X_{r3}(t))*[f(X_{r3}(t))-f(G(t))]/D$$

$$D=f_{high}-f_{low}$$

参考点 $X_{r1}(t)$，$X_{r2}(t)$，基点 $X_{r3}(t)$ 为任意选取的其他三个不同的粒子，即 $i\neq r1\neq r2\neq r3$。

式（5）采用了贪婪选择机制更新粒子 $X_i(t)$，即，仅仅当候选粒子 $T_i(t)$ 的评估值 $f(T_i(t))$ 小于粒子 $X_{r3}(t)$ 的评估值 $f_i(t)$ 时，用候选粒子替换掉 $X_i(t)$。式（4）是利用基点 $X_{r3}(t)$ 以及其与参考点之间的信息共享来构造候选粒子 $T_i(t)$。

F_0 记录了参考点 $X_{r1}(t)$ 与基点 $X_{r3}(t)$ 之间的交流信息。当 $f_{r1}(t)<f_{r3}(t)$ 时，$S_0\triangleq[f(X_{r3}(t))-f(X_{r1}(t))]/D>0$，$F_0$ 将引导 $X_{r3}(t)$ 向 $X_{r1}(t)$ 方向搜索，而且，评估值差异越大，在方向 $(X_{r1}(t)-X_{r3}(t))$ 上搜索步长 S_0 就越多，反之，差异越小，S_0 就越多；当 $f_{r1}(t)\geq f_{r3}(t)$ 时，$S_0<0$，F_0 将引导 $X_{r3}(t)$ 在其邻域内搜索，这时搜索步长 S_0 刻画了邻域的大小，而且，评估值差异越大，邻域 S_0 就越小，反之，差异越小，邻域就越大。因此，通过这种方式，一方面可以很好地平衡局部搜索和全局探索之间的矛盾，另一方面算法自适应地在加速收敛和提高种群多样性之间转换。

同样的处理思路，F_1 记录了参考点 $X_{r2}(t)$ 的个体经历最优 $g_{r2}(t)$ 与基点 $X_{r3}(t)$ 之间的信息。

因为必有 $f_{low}(t)\leq f_{r3}(t)$，所以 F_2 引导基点 $X_{r3}(t)$ 向种群全局最优 $G(t)$方向搜索。

这种引入两个粒子间的信息交流机制类似于电磁机制。把每个粒子都当作一个点电荷，函数值越好（小）点电荷电量越高，对其他粒子的吸引力越强。一个粒子的下一步搜索方向是其他粒子对它施加的"合力"，因此这种机制不仅仅有利于改善算法性能[12,21]，而且能够达到免参数目的。

3.2 算法描述

a)初始化种群，种群规模 POP=3n+1，变异系数 Cr=0.1，$t=0$；

b)排序。计算函数值 $f_i(t)$，并按照函数值从小到大进行排序；

c)单纯形搜索。将前 n+1 个最好的粒子作为单纯形顶点，执行 NM 算法，改进 n+1 个顶点中最差粒子；

d)粒子群优化。以剩余的 2n 个粒子构成粒子种群 $P(t)$，对每一个粒子 $X_i(t)\in P(t)$，

(a)按照式（4）计算候选粒子 $T_i(t)=X_{r3}(t)+F_0+F_1+F_2$；

(b)边界检查。对 $\forall j$，如果 $T_{i,j}(t)<LB$，则 $T_{i,j}(t)=LB$；如果 $T_{i,j}(t)>UB$，则 $T_{i,j}(t)=UB$；

(c)贪婪选择。如果 $f(T_i(t))$ < $f_i(t)$，则 $X_i(t+1)=T_i(t)$，否则 $X_i(t+1)=X_i(t)$；

(d)变异。对 $\forall j$，如果 rand > Cr，则 $X_{i,j}(t+1)=X_{i,j}(t)$；

e)如果满足终止条件，则停机；否则，$t=t+1$，转步骤 b)。

为了增加种群的多样性，避免算法过早陷入局部最优，在步骤 c 中，NM 算法只执行一步迭代，步骤 d)中，增加了变异操作。

4　数值实验

4.1　对照算法介绍

本文将比较包括本文算法在内的五个算法的数值实验结果。首先介绍其他四种对照算法。

1）惯性权重递减粒子群算法（degressive inertia weight particle swarm optimization，DPSO）

在标准粒子群算法中，惯性权重 w 对 $V_{i,j}(t)$，即该粒子上次速度更新的影响力相比另外两个分量占据主要位置，所以，当 w 值较大时，$V_{i,j}(t)$ 的范数较大，每次迭代粒子的“前进距离”较长，有利于全局搜索以找到最优解所在的邻域。反之，当 w 较小时，三个分量对迭代方向影响力相近，使算法在一个相对较小的区域内能够更精确地搜索，即加强了局部搜索的能力[5]。因此，如果使惯性权重 w 能够随迭代次数递减，则算法在运行前期全局搜索能力加强，在运行后期局部搜索能力加强，能够使算法整体性能提高。惯性权重 w 的递减公式如下：

$$w = w_{max} - \left(\frac{iter}{iter_{max}}\right) \times (w_{max} - w_{min})$$

其中：w_{max}，w_{min} 分别为设定的 w 的最大值和最小值，Shi 等人推荐这两个参数的值分别为 1.4 和 0[7]，$iter$，$iter_{max}$ 分别为当前迭代次数和最大迭代次数。

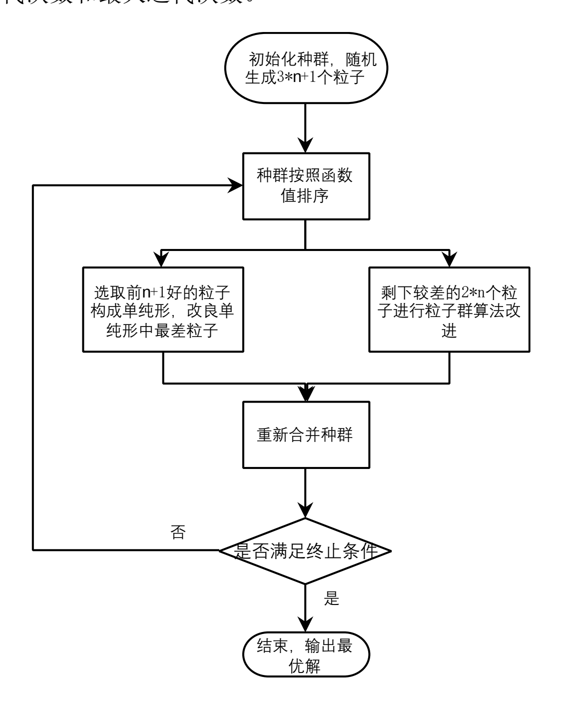


图 5　NM-PSO 流程图

Fig.5　Flowchart of NM-PSO

采用此算法作为比较算法之一的原因是，此算法也采用自适应参数的机制，只是这个自适应机制较为简单，NM-PSO 的自适应机制较为复杂，经过对比能够显现出算法在使用自适应参数机制由简单到精密的性能提升。

2）随机惯性权重粒子群算法（random inertia weight particle swarm optimization，RPSO）

RPSO 是将 PSO 中的惯性权重 w 改为[0.5,1]中的均匀分布随机数，即 $w=rand$（0.5,1）[5]。其余步骤均与 PSO 保持一致。

3）混合单纯形粒子群算法[16]（hybrid simplex particle swarm optimization，HSPSO）

HSPSO 算法在每一次迭代中，一方面利用单纯形法对最好的前 N+1 个粒子进行更新，另一方面利用粒子群优化算法对剩余的 2N 粒子进行改进。在单纯形法每一次执行中，只需迭代一次，使用新的粒子替换原单纯形的最差顶点，这样不仅可以大大降低算法的计算复杂度，而且有效地保持了算法的求解性能[16]。

4）单纯形粒子群算法[19]（simplex method particle swarm optimization，SMPSO）

在粒子群优化算法产生随机初始值后，使用单纯形法对初始值进行处理，使用处理后的初始值进行粒子群优化算法的后续步骤。经过单纯形法处理后的初始值如果已经接近全局最优解，则经过粒子群算法的迭代会很快收敛到全局最优解，即使处理后的初始值陷入了局部最优解，粒子群算法也会有很大的概率使得结果跳出局部最优解。这种处理方法可以有效发挥这两种算法的各自优势[19]。

4.2　数值实验数据

五种算法分别对 10 个 30 维测试函数（见附录）独立进行 50 次数值实验，数值结果见表 1-4，图 4.2.1-10，各算法的参数设置如下：

1）DPSO

POP=3*30+1，c_1=c_2=1.494，w_{max}=1.4，w_{min}=0

2）RPSO

POP=3*30+1，c_1=c_2=1.494，w=$rand$(0.5,1)

3）HSPSO[16]

POP=3*30+1，c_1=0.6，c_2=1.6，w=$rand$(0.5,1)，

变异系数 lambda=0.85 反射系数 r=1.75 扩展系数 e=2.75，压缩系数 c=0.75

4）SMPSO[19]

POP=3*30+1，c_1=1.5，c_2=2.5，w=$rand$(0.5,1)

表 1~3 分别记录了每个算法 50 次运行的最好解、最差解、平均值与理论准确解之间的误差；表 4 记录了每个算法 50 次运行结果的方差。图 6-15 反映了 50 次运行结果的平均值随评价次数变化的情况。为了图形对比明显，对纵坐标取对数变换。

表 1　最好解与准确解的误差

Table 1　Error between the best solution and the exact solution

函数名	DPSO	HSPSO	NM-PSO	RPSO	SMPSO
Ackley	1.80E-12	1.82E-07	2.52E-09	1.31E-05	1.8E-06
Sphere	6.72E-20	1.74E-10	2.36E-13	3.46E-07	1.17E-09
Rosenbrock	0.591478	7.632082	10.0137	4.090019	4.31444
Rastrigin	25.86892	14.00768	4.72E-12	18.9097	8.954632
Griewank	6.51E-14	3.52E-07	5.7E-08	0.000207	1.47E-06
Styblinski-tang	28.27347	70.68362	2.89E-05	0.7069	18.24352
Levy	8.34E-15	4.01E-09	1.09E-11	1.83E-05	1.21E-08

Exponential	3.33E-16	0	1.11E-16	8.43E-14	5.55E-16
Zakharov	2.47734	0.00044	0.412443	49.60656	5.132545
Schwefel 2.22	8.42E-05	1.41E-05	3.55E-08	4.03E-05	4.93E-07

表 2　最差解与准确解的误差

Table 2　Error between the worst solution and the exact solution

函数名	DPSO	HSPSO	NM-PSO	RPSO	SMPSO
Ackley	3.734012	1.98E-06	4.11E-08	0.003062	0.00088
Sphere	6.95E-13	1.06E-08	1.36E-11	4.3E-05	1.79E-06
Rosenbrock	323.1116	3.03E+03	134.3423	530.7808	118.3099
Rastrigin	94.52081	85.56606	3.054697	77.60666	39.79833
Griewank	0.08128	0.071211	0.021834	0.056881	0.061479
Styblinski-tang	226.1875	240.3243	14.13675	169.8959	169.6432
Levy	6.522771	6.546213	1.35E-09	4.365262	0.003875
Exponential	9.33E-15	4.33E-15	4.44E-16	2.65E-11	1.39E-13
Zakharov	95.01998	126.6204	13.16558	1118.755	96.28544
Schwefel 2.22	0.097455	6.327632	1.51E-07	0.002097	1.14E-05

表 3　平均解与准确解的误差

Table 3　Error between the mean solution and the exact solution

函数名	DPSO	HSPSO	NM-PSO	RPSO	SMPSO
Ackley	0.95083	6.13E-07	1.03E-08	0.000165	5.63E-05
Sphere	2.13E-14	3.1E-09	2.02E-12	6.98E-06	2.19E-07
Rosenbrock	57.4121	231.3918	59.97354	64.70934	41.96105
Rastrigin	55.61811	44.35636	0.590349	41.96631	17.85171
Griewank	0.015881	0.015977	0.000619	0.012978	0.013808
Styblinski-tang	138.5399	142.4982	0.346304	90.87972	77.35274
Levy	2.085249	1.308247	2.15E-10	0.413019	7.97E-05
Exponential	1.31E-15	3.82E-16	1.18E-16	2.8E-12	2.7E-14
Zakharov	17.26885	26.82361	4.285058	402.4986	34.16138
Schwefel 2.22	0.009718	0.126662	8.22E-08	0.000425	2.73E-06

表 4　数值解的方差

Table 4　Variance of the solution

函数名	DPSO	HSPSO	NM-PSO	RPSO	SMPSO
Ackley	0.808365	1.34E-13	6.14E-17	1.86E-07	2.09E-08
Sphere	9.93E-27	5.46E-18	5.21E-24	8.76E-11	1.81E-13
Rosenbrock	4878.856	5.21E+05	1000.014	7922.594	796.9568
Rastrigin	203.6541	373.4897	0.651505	173.3405	34.57248
Griewank	0.00038	0.000301	9.68E-06	0.000178	0.000284
Styblinski-tang	1631.403	1426.172	4.162142	1175.893	997.812
Levy	2.415501	1.831068	9.04E-20	0.545345	3E-07
Exponential	1.96E-30	4E-31	2.22E-33	2.22E-23	1.07E-27
Zakharov	378.6455	1181.804	8.551882	62813.37	480.7299
Schwefel 2.22	0.00041	0.80075	8.06E-16	2.33E-07	4.82E-12

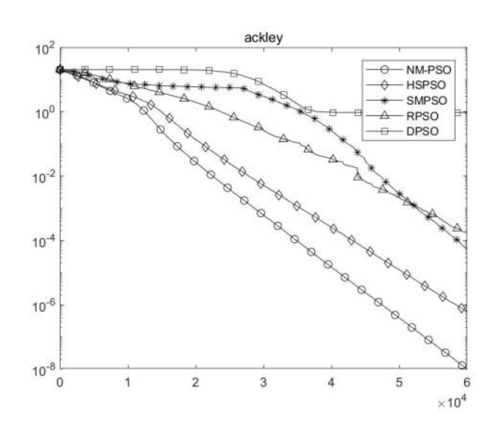


图 6　Ackley 函数平均值随评估次数的变化

Fig.6　Variations of mean values with evaluations for Ackley function

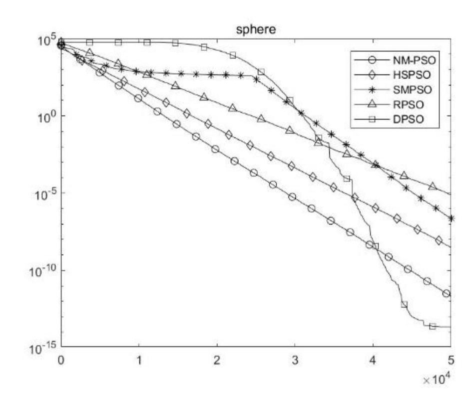


图 7　Sphere 函数平均值随评估次数的变化

Fig.7　Variations of mean values with evaluations for Sphere function

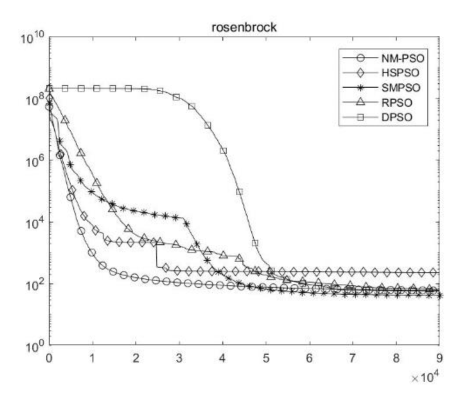


图 8　Rosenbrock 函数平均值随评估次数的变化

Fig.8　Variations of mean values with evaluations for Rosenbrock function

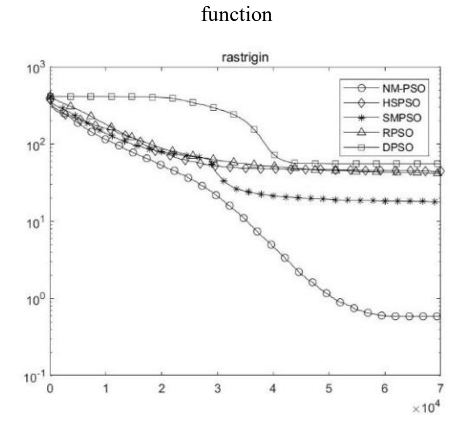


图 9　Rastrigin 函数平均值随评估次数的变化

Fig.9　Variations of mean values with evaluations for Rastrigin function

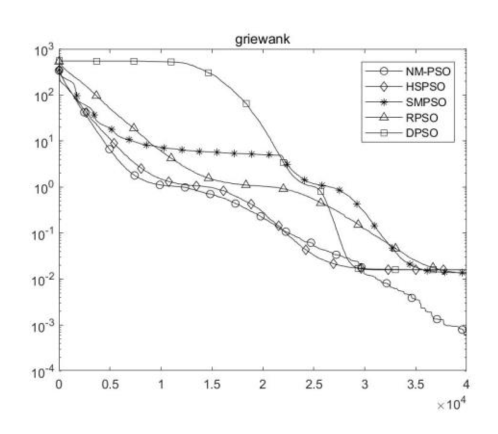


图 10　Griewank 函数平均值随评估次数的变化

Fig.10　Variations of mean values with evaluations for Griewank function

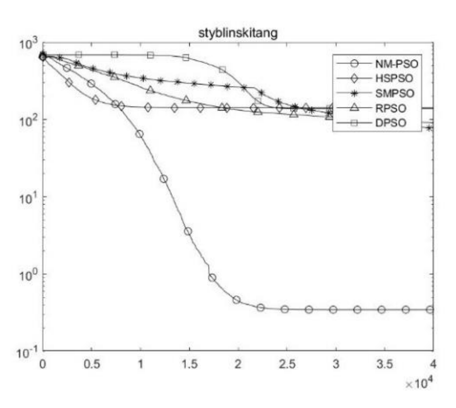


图 11　Styblinski-Tang 函数平均值随评估次数的变化

Fig.11　Variations of mean values with evaluations for Styblinski-Tang function

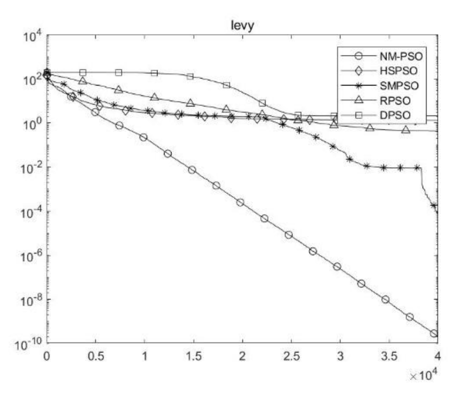


图 12　Levy 函数平均值随评估次数的变化

Fig.12　Variations of mean values with evaluations for Levy function

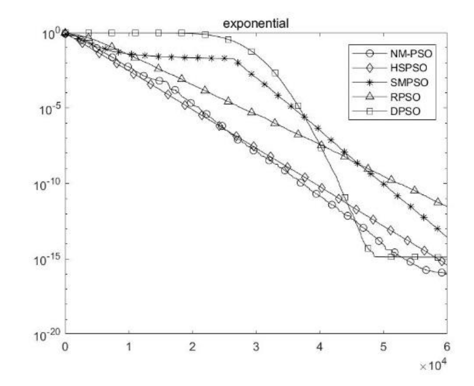
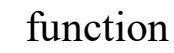


图 13　Exponential 函数平均值随评估次数的变化

Fig.13　Variations of mean values with evaluations for Exponential


function

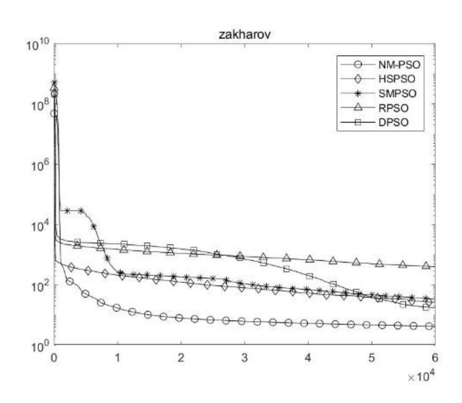


图 14　Zakharov 函数平均值随评估次数的变化

Fig.14　Variations of mean values with evaluations for Zakharov function

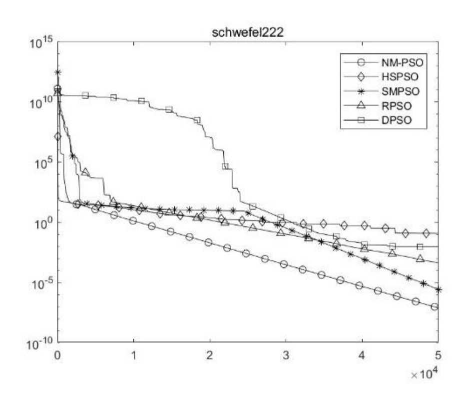


图 15　Schwefel 2.22 函数平均值随评估次数的变化

Fig.15　Variations of mean values with evaluations for Schwefel 2.22 function

4.3　数值实验分析

4.3.1 表格数据分析

观察表 4，NM-PSO 在计算数值结果的方差上明显优于其他算法，仅对测试函数 sphere 上方差略大于 DPSO，表明 NM-PSO 在运行的稳定性和结果的稳定性上明显高于其他算法。观察表 3，NM-PSO 在运算结果的平均误差方面也处于总体领先，对大部分测试函数，NM-PSO 领先其他算法多个数量级，仅对测试函数 rosenbrock 运算的平均值与其他算法差距不大，说明 NM-PSO 对运算结果的平均误差优于其他算法，算法的稳定性强，结果也更精确。对表 1 和 2，对测试函数 Rastrigin、Styblinski-Tang、Exponential 和 schwefel2.22，NM-PSO 的计算结果优于其他所有算法，但对测试函数 Rosenbrock，NM-PSO 表现不佳，这可能是因为测试函数具有某些特殊的性质，观察表 2 可以得出在 8 个测试函数的最差解结果中，本文算法都处于领先地位，特别在 Rastrigin、Styblinski-Tang、Levy、Exponential、Zakharov、Schwefel 2.22 上领先了 1 个数量级及以上，尤其 levy 函数达到了 9 个数量级，体现 NM-PSO 跳出局部最优的能力，可以看出本算法的稳定性是远远好于其他算法的。对表格的分析总体表明，NM-PSO 在绝大多数测试函数下性能表现最佳，即使对表现不佳的测试函数，性能也不会处于完全劣势，所以 NM-PSO 求解能力总体处于领先水平。



4.3.2 图表数据分析

对测试函数 Rastrigin、Styblinski-Tang、Levy 本文算法 NM-PSO 在平均求解精度上相比其他四个算法表现具有压倒性的优势。并且 NM-PSO 在跳出局部最优解方面展示出了强大的能力，在进行了大约 20000~40000 次目标函数调用后，其余四个算法陷入了局部最优解，在之后的调用中，最优解基本没有再改进，反观本算法，最终的最优值要好上 102~104 数量级。对于测试函数 Ackley、Griewank、Exponential、Zakharov、Schwefel 2.22，本算法在平均求解精度对其他算法虽然没有领先太多，但收敛速度（纵坐标下降速度）全都好于其余四个算法，在同样的函数评估次数下，只有 HSPSO 算法在 Griewank 测试的前半段性能与本算法不相上下，其余情况 NM-PSO 算法皆能够取得最好效果。对于测试函数 sphere，三个基于单纯形的粒子群算法效果均好于 RPSO 算法，但比 DPSO 算法差，这可能是单纯形与粒子群算法融合结构决定的。在对 Rosenbrock 算法进行数值实验时，五个算法均未能很好的达到全局最优，误差都在 102 数量级。更具体地，对比同样基于单纯形的算法 HSPSO，SMPSO，测试函数 Ackley 表明，三个函数在最终结果，收敛速度方面均要好于传统粒子群算法，且其中本文算法依旧是最优。对于函数 Rastrigin、Griewank、Levy、Exponential、Zakharov 三个算法中总有两者要好于传统算法，体现了单纯形搜索带来的好处。进一步比较三者，本文算法 NM-PSO 更是有三者中最佳的性能。

通过上述对 10 个测试函数的数值实验分析得出结论：在大部分测试函数下，NM-PSO 算法总体上求解精度较高，性能表现优越。

5 结束语

为了提高粒子群优化算法的求解性能，本文一方面借鉴电磁机制原理设计了一种 PSO 迭代方式，另一方面利用单纯形搜索来加速算法收敛。数值实验表明，具有免参数特征的 NM-PSO 能够很好地平衡局部搜索和全局探索之间的矛盾。借鉴传统优化方法的成熟理论、技巧来指导设计进化算法是本文的初衷，如何设计一种更好、更简洁的算法融合方式是本文今后研究的重点。

参考文献：

[1] Kennedy J. Particle swarm optimization [M]// Encyclopedia of machine learning. Boston: Springer, 2011: 760-766.

[2] Clerc M, Kennedy J. The particle swarm-explosion, stability, and convergence in a multidimensional complex space [J]. IEEE Trans on Evolutionary Computation, 2002, 6 (1): 58-73.

[3] Kennedy J. Swarm intelligence [M]// Handbook of nature-inspired and innovative computing. Boston: Springer, 2006: 187-219.

[4] Shi Yuhui, Eberhart R. A modified particle swarm optimizer [C]// Proc of IEEE International Conference on Evolutionary Computation, IEEE World Congress on Computational Intelligence. Piscataway, NJ: IEEE Press, 1998: 69-73.

[5] Eberhart R C, Shi Yuhui. Tracking and optimizing dynamic systems with particle swarms [C]// Proc of Congress on Evolutionary Computation. 2001: 94-100.

[6] Shi Yuhui, Eberhart R C. Parameter selection in particle swarm optimization [C]// Proc of International Conference on Evolutionary Programming. Berlin: Springer, 1998: 591-600.

[7] Eberhart R C, Shi Yuhui. Comparison between genetic algorithms and particle swarm optimization [C]// Proc of International conference on evolutionary programming. Berlin: Springer, 1998: 611-616.

[8] Eberhart R C, Shi Yuhui. Comparing inertia weights and constriction factors in particle swarm optimization [C]// Proc of Congress on Evolutionary Computation. 2000: 84-88.

[9] 阳春华，钱晓山，桂卫华. 一种混沌差分进化和粒子群优化混合算法 [J]. 计算机应用研究, 2011, 28 (2) . (Yang Chunhua, Qian Xiaoshan, Gui Weihua. A hybrid algorithm for chaotic difference evolution and particle swarm optimization [J] Application Research of Computers, 2011, 28 (2)

[10] 李洪亮，侯朝桢，周绍生. 一种高效的改进粒子群优化算法 [J]. 计算机工程与应用, 2008, 44 (1): 14-16. (Li Hongliang, Hou Chaozhen, Zhou Shaosheng. An efficient improved particle swarm optimization algorithm [J]. Computer Engineering and Applications, 2008, 44 (1): 14-16)

[11] Kennedy J. Bare bones particle swarms [C]// Proc of Swarm Intelligence Symposium. 2003: 80-87.

[12] Birbil Ş İ, Fang S C. An electromagnetism-like mechanism for global optimization [J]. Journal of Global Optimization, 2003, 25 (3): 263-282.

[13] Loengarov A, Tereshko V. A minimal model of honey bee foraging [C]// Proc of IEEE Swarm Intell. Symp. 2006: 175-182.

[14] Marini F, Walczak B. Particle swarm optimization (PSO): a tutorial [J]. Chemometrics and Intelligent Laboratory Systems, 2015, 149: 153-165.

[15] Zahara E, Kao Y T. Hybrid Nelder–mead simplex search and particle swarm optimization for constrained engineering design problems [J]. Expert Systems with Applications, 2009, 36 (2): 3880-3886.

[16] Fan S K S, Zahara E. A hybrid simplex search and particle swarm optimization for unconstrained optimization [J]. European Journal of Operational Research, 2007, 181 (2): 527-548.

[17] Hsu C C, Gao C H. Particle swarm optimization incorporating simplex search and center particle for global optimization [C]// Proc of IEEE Conference on Soft Computing in Industrial Applications. 2008: 26-31.

[18] 任小康，郝瑞芝，孙正兴，等. 基于单纯形法的量子粒子群优化算法 [J]. 微电子学与计算机, 2010 (1): 154-157. (Ren XiaoKang, Hao Ruizhi, Sun Zhengxing, *et al.* Quantum particle swarm optimization algorithm based on simplex method [J] Microelectronics and Computer Science, 2010 (1): 154-157.)

[19] 武可栋，韦增欣，杨天山. 一种使用单纯形法优化的粒子群算法 [J]. 数学的实践与认识, 2018, 48 (7): 199-205. (Wu Kedong, WeiZengxin,

Yang Tianshan. A particle swarm optimization algorithm using simplex method [J]. Journal of Mathematics in Practice and Theory, 2018, 48 (7): 199-205.)

[20] Conn A R, Scheinberg K, Vicente L N. Introduction to derivative-free optimization [M]// MOS-SIAM Series on Optimization, 2009: 141-162

[21] Trelea I C. The particle swarm optimization algorithm: convergence analysis and parameter selection [J]. Information processing letters, 2003, 85 (6): 317-325.

附录：

下面表格列出了所有测试函数的维数，取值范围、最优值（表 5）以及解析式（表 6）。

表 5　10 个测试函数信息

Table 5　Information of 10 Test Functions

函数名	维数	取值范围	最优值
Ackley	30	[-32,32]	0
Sphere	30	[-100,100]	0
Rosenbrock	30	[-30,30]	0
Rastrigin	30	[-5.12,5.12]	0
Griewank	30	[-600,600]	0
Styblinski-tang	30	[-5,5]	-1174.9797
Levy	30	[-10,10]	0
Exponential	30	[-1,1]	-1
Zakharov	30	[-5,10]	0
Schwefel 2.22	30	[-10,10]	0

表 6　测试函数

Table 6　Test Functions

函数名	解析式
Ackley	$f(x)=-20*exp\left(-0.2\sqrt{\frac{1}{n}\sum_{t=1}^{n}x_i^2}\right)-exp\left(\frac{1}{n}\sum_{i=1}^{n}cos(\pi x_i)\right)+20+exp(1)$
Sphere	$f(x)=\sum_{i=1}^{n}x_i^2$
Rosenbrock	$f(x)=\sum_{i=1}^{n-1}[100(x_{i+1}-x_i^2)^2+(x_i-1)^2]$
Rastrigin	$f(x)=\sum_{i=1}^{n}[x_i^2-10cos(2\pi x_i)]+10n$
Griewank	$f(x)=1+\sum_{i=1}^{n}\frac{x_i^2}{4000}-\prod_{i=1}^{n}cos\left(\frac{x_i}{\sqrt{i}}\right)$
Styblinski-tang	$f(x)=\frac{1}{2}\sum_{i=1}^{n}(x_i^4-16x_i^2+5x_i)$
Levy	$f(x)=sin^2(\pi w_1)+\sum_{i=1}^{n-1}(w_i-1)^2[1+10sin^2(\pi w_i+1)]+(w_n-1)^2[1+sin^2(2\pi w_n)]$ where $w_i=1+\frac{x_i-1}{4}, i=1,\cdots,n$
Exponential	$f(x)=-exp\left(-0.5\sum_{i=1}^{n}x_i^2\right)$
Zakharov	$f(x)=\sum_{i=1}^{n}x_i^2+\left(\sum_{i=1}^{n}0.5ix_i\right)^2+\left(\sum_{i=1}^{n}0.5ix_i\right)^4$ where $i^2=-1$
Schwefel 2.22	$f(x)=\sum_{i=1}^{n}\lvert x_i\rvert+\prod_{i=1}^{n}\lvert x_i\rvert$